



# TwinOps – DevOps meets Model-Based Engineering and Digital Twins for the engineering of CPS

Jerome Hugues, Anton Hristosov, John J. Hudak  
and Joe Yankel

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Copyright 2020 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

DM20-0953

# Model-Based Engineering for Cyber-Physical Systems@SEI



Create the best design  
that holds up over time  
as the system evolves.



Test the design without  
having to write any code.



Build a single model to assess  
hardware and embedded software  
before the system is built.

## SAE AADL / ACVIP

- Standardized language and process for the engineering safety-critical systems.

## OSATE

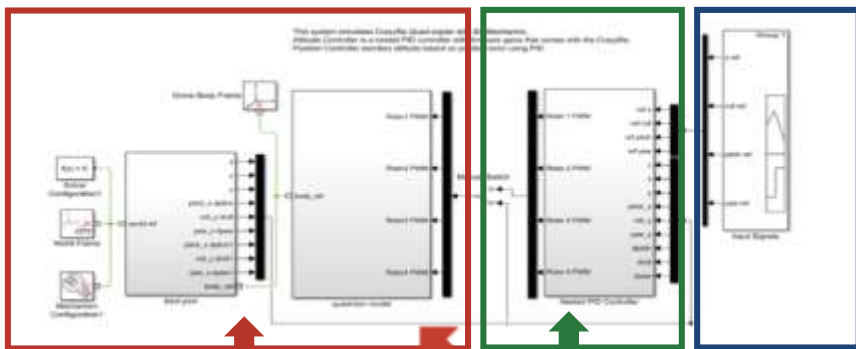
- Open Source AADL toolset for performing verification and validation (V&V).

## Pilot Projects

- Maturity increased through case studies and feedback from practitioners

# TwinOps problem space: CPS Integration and Testing

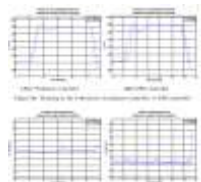
High-Level Architecture



TwinOps: leverage other source of truth  
(e.g., CAD, Physics) to improve SW V&V  
⇒ Use precise models instead of (naïve) abstractions for improved SW V&V  
⇒ Combine domains, including SysEng

Implementation Space

SW/HW Architecture

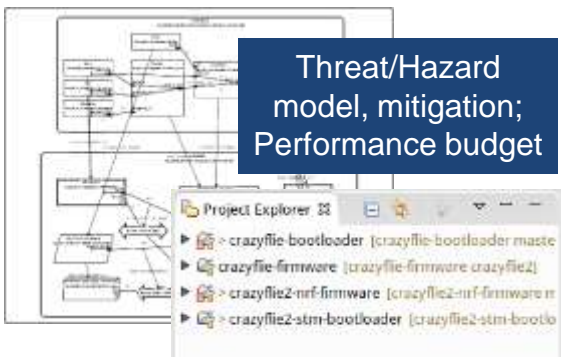


?

Threat/Hazard  
model, mitigation;  
Performance budget



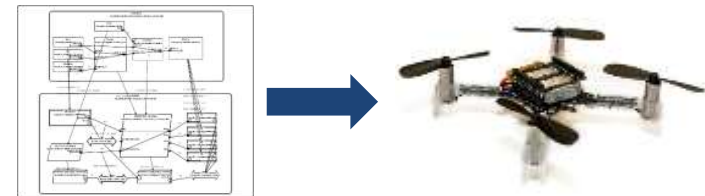
Actual sources



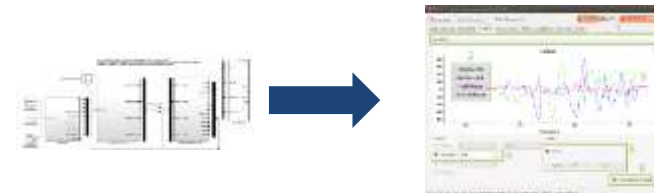
Mismatch, late  
discovery and rework

# Technology Focus: Models and Code Generation

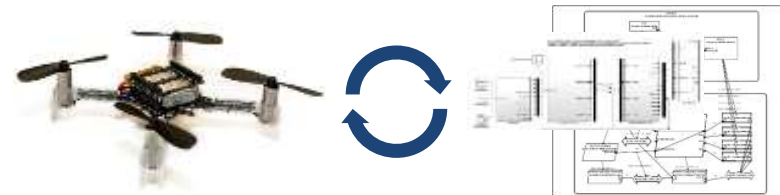
One can **generate code** from models ready to be embedded in the system (e.g., AADL to C) and get insights from the system to refine the model metrics.



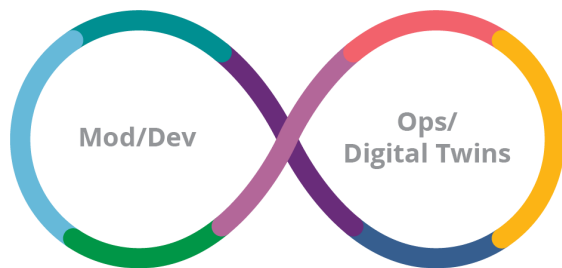
One can **simulate models** and generate simulation code as a mock-up of some system parts.



One can build **Digital Twins**, that compare actual system and its digital simulated doppelganger.



# From DevOps to ModDevOps



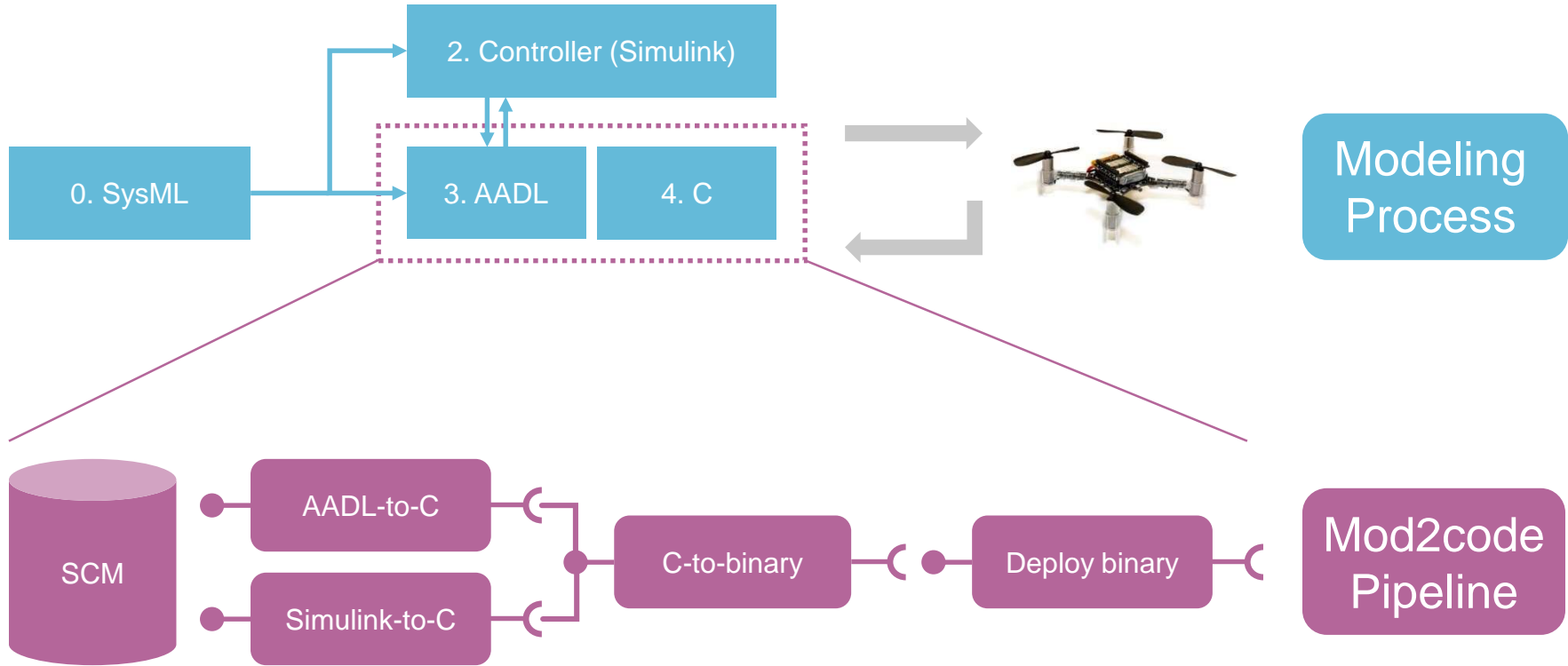
DevOps delivers software faster with increased quality:

- Continuous integration/deployment
- Containerized systems

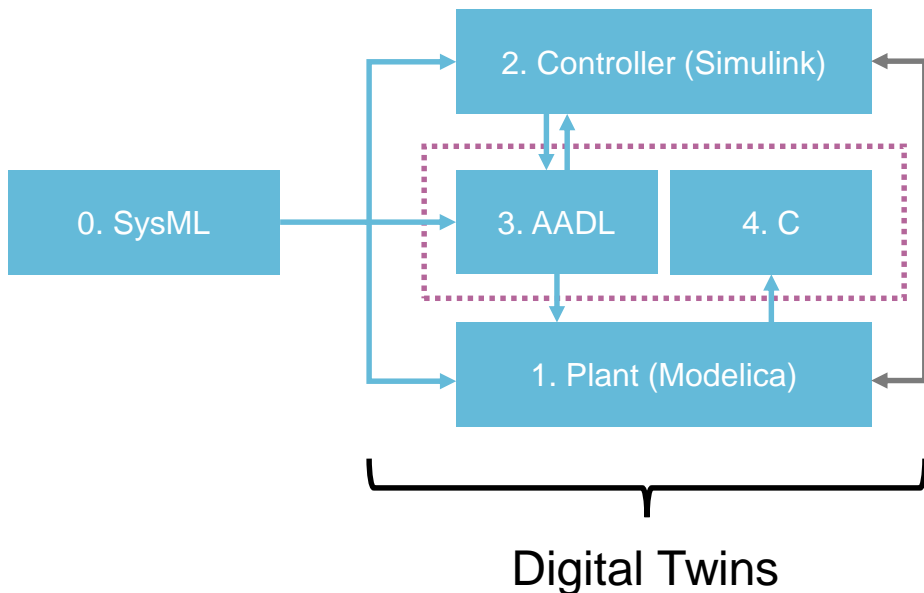
DevOps is a software process, to be adapted to systems.

**ModDevOps** is a ~~software~~*systems/software* co-engineering culture and practice that aims at unifying *systems engineering (Mod)*, software development (Dev) and software operation (Ops). The main characteristic of **ModDevOps** is to strongly advocate *abstraction*, automation and monitoring at all steps of *system* construction, from integration, testing, releasing to deployment and *infrastructure* management. (adapted from <https://software.af.mil/training/devops/> )

# ModDevOps in Action – Modeling Process



# From ModDevOps to TwinOps



1-2-3-4: “mega-modeling” V&V

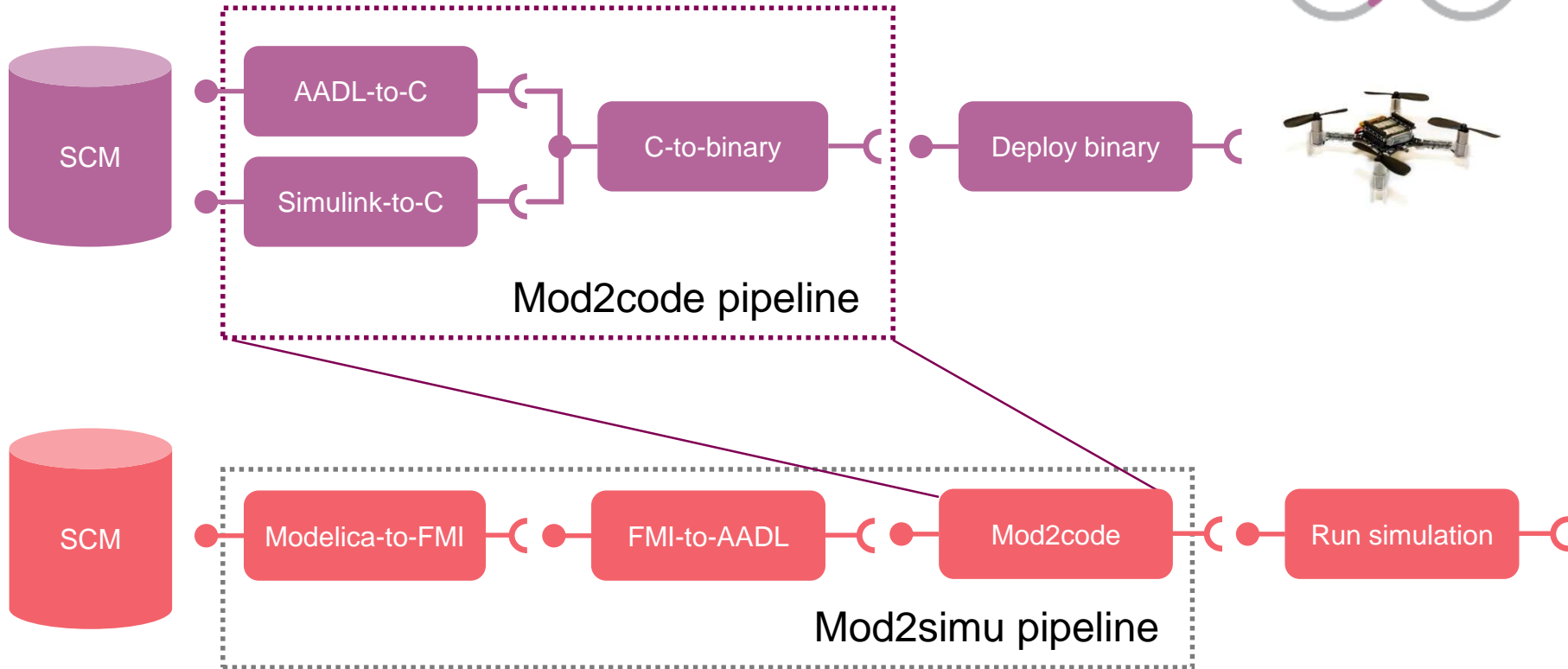
- 1-2: HLR validation
- 2-(3+4): validation of LLR
- 1+(3+4): virtual integration



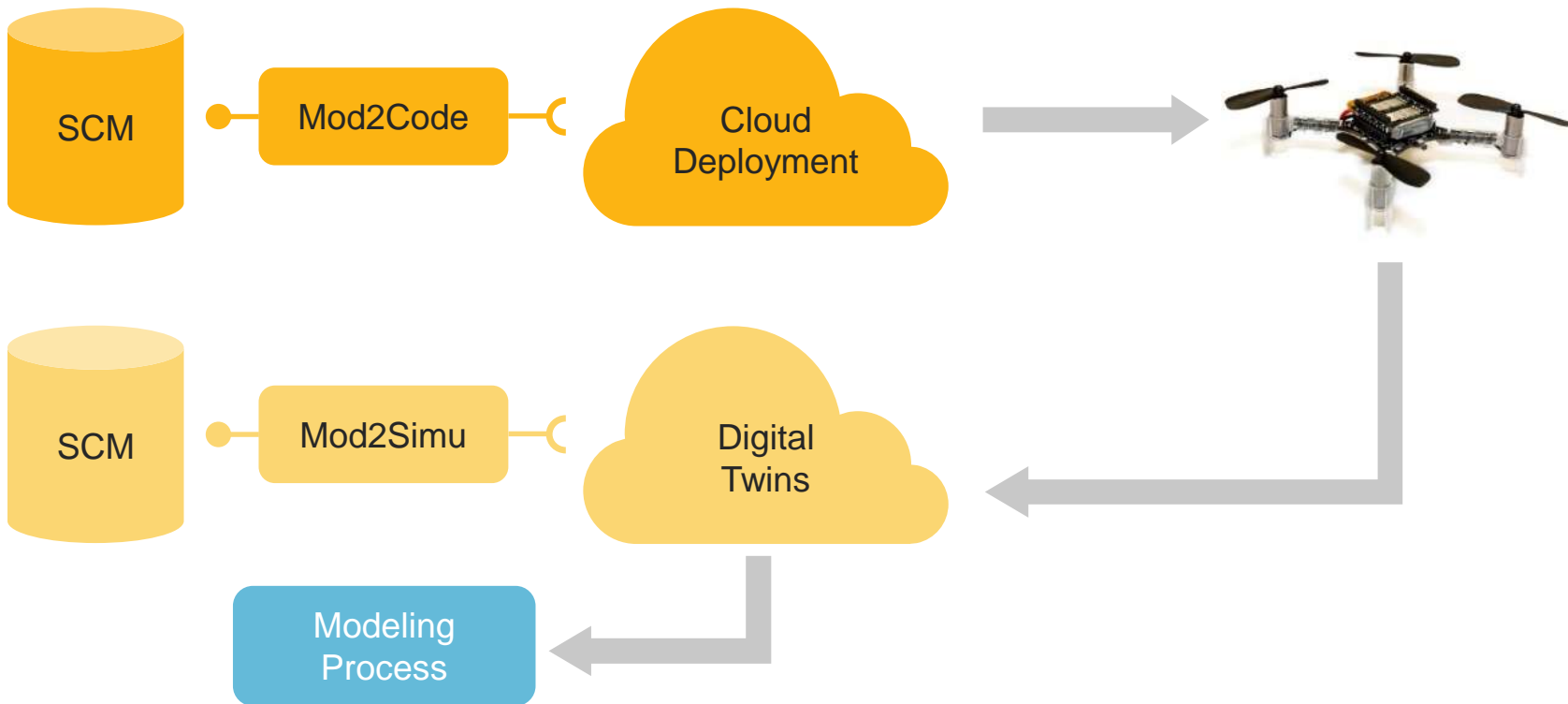
Digital Twins of UAV vs. UAV flying: validation of Modelica model, efficiency of the controller (overshoot verification) and timing verification of software.



# ModDevOps in Action – ModDevOps Pipeline #2



# From ModDevOps to TwinOps



# TwinOps: Continuous System Improvement through ModDevOps and Digital Twins

